AW-CU300A V3

IoT Connectivity Module for AWS IoT Core

Getting Started Guide

Rev. 0.1

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

Revision History

Version	Revision Date	Description	Initials	Approved
0.1	2021/01/04	Initial version	Renton Tao Jackson Boon	N.C. Chen S.C Chueh

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

Table of Contents

1. Introduction of AW-CU300A V3 4

1.1 Product Overview 4
1.2 Block Diagram 5
1.3 Schematics 6

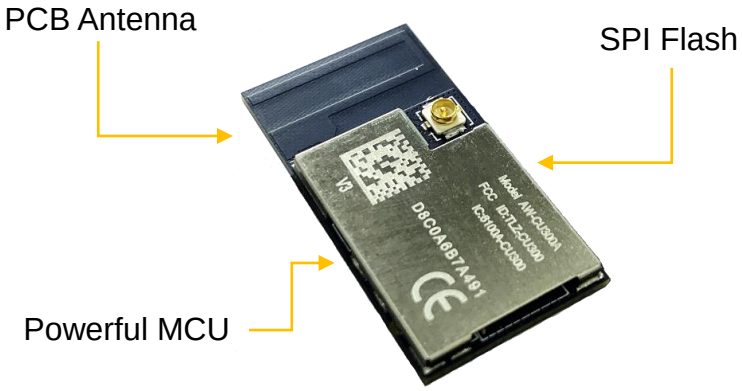
2. AWS Command Example 7

2.1 Getting Started with AWS IoT Core 7
2.2 Publish and Monitor MQTT message on the cloud 11

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

1. Introduction of AW-CU300A V3

1.1 Product Overview

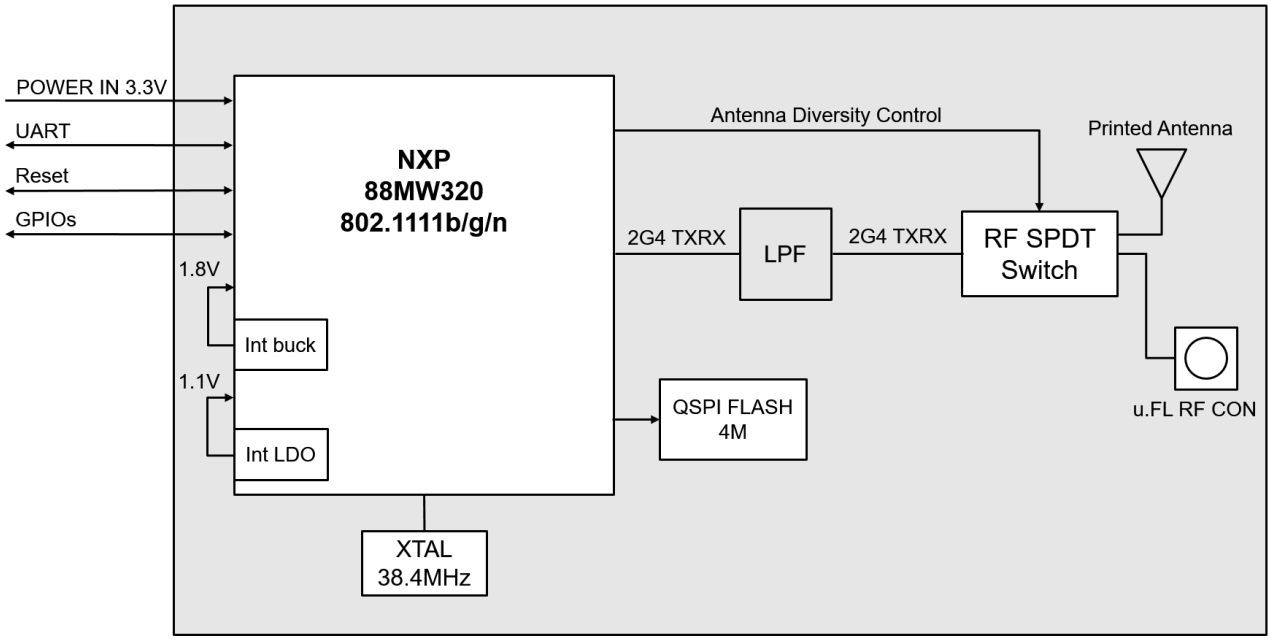


AW-CU300A V3 is a Wi-Fi Module with FreeRTOS qualified MCU that uses AT commands to securely and efficiently communicate with AWS IoT Core

- Hardware specification defined by Amazon and AzureWave
- With AICM, end-device become AWS IoT Device
- UART interface for end-device to connect with
- Rich AT commands for end-device to communicate with AWS IoT Core

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

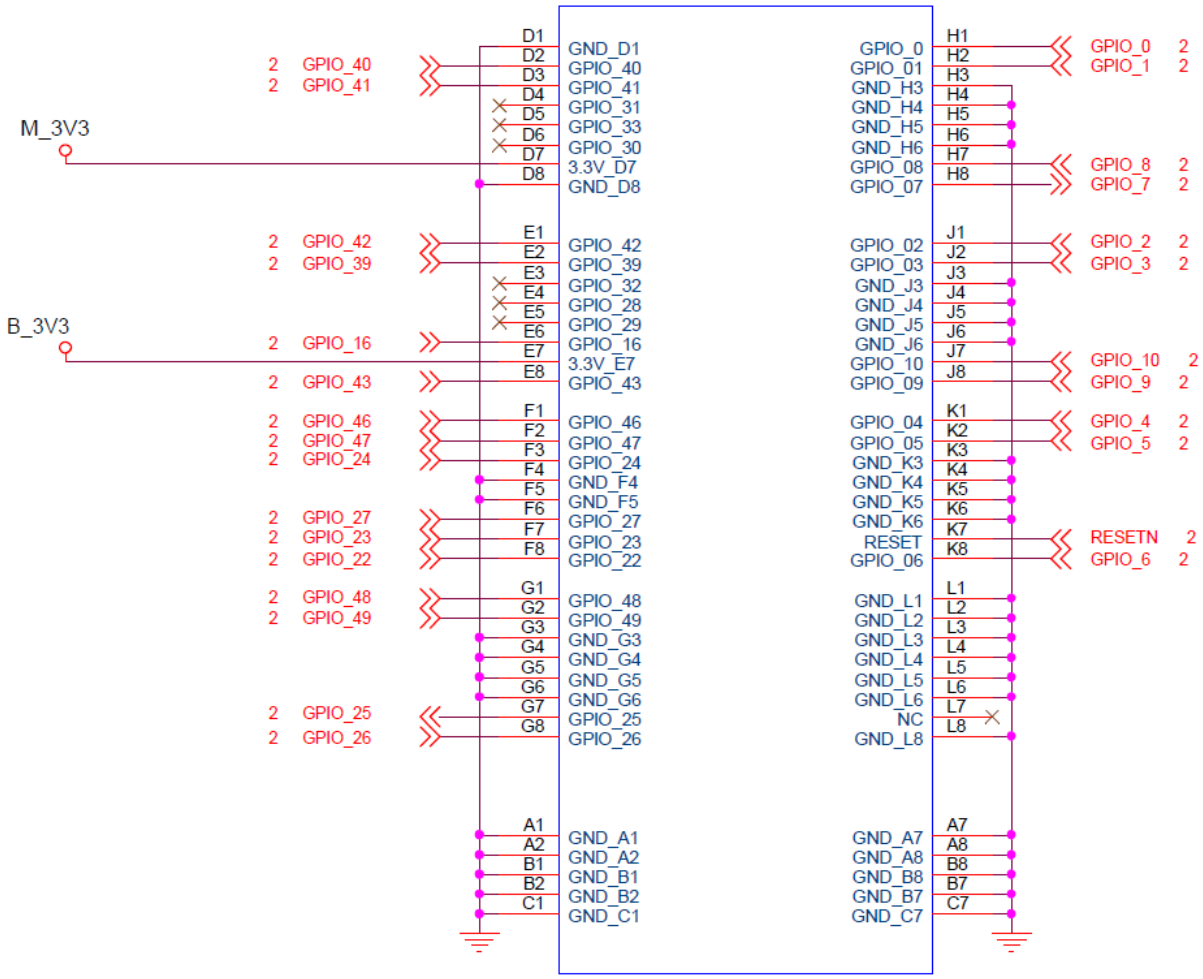
1.2 Block Diagram



The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

1.3 Schematics

Module pinout for AW-CU300A V3



The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

2. AWS Command Example

Please find the command details in **AWS CONNECTOR AT Command Set**.

Below are commands for the demo:

1. Turn on Wi-Fi module: **AT+WIFI_On**
2. Retrieve the Wi-Fi AP | Station Mode: **AT+WIFI_GetMode**
3. Perform a Wi-Fi network scan: **AT+WIFI_Scan**
4. Set and store the Wi-Fi AP information when AW-CU300A V3 in Station Mode:

   AT+WIFI_SetAP=**SSID**,**password,security type**
   SSID: SSID of AP (case sensitive)
   password: password for AP (case sensitive)
   security type: OPEN | WEP | WPA | WPA2

5. Connect to the AP: **AT+WIFI_Connect**
6. Define and store Thing-specific configuration:

   AT+THING_Set=client ID,endpoint,client certificate,client private key
   client ID: Thing name(Client ID)
   endpoint: AWS IoT endpoint URL
   client certificate: Certificate for this Thing
   client private key: Private key for this Thing

7. Connect the client to MQTT broker: **AT+MQTT_Connect**
8. Subscribe to and save MQTT topic: **AT+MQTT_Subscribe**=<topic>,<qos>
9. Publish to MQTT topic:

   AT+MQTT_Publish=topic,message,qos
   topic: Topic to publish to
   message: Message to publish
   qos: 0 | 1

2.1 Getting Started with AWS IoT Core

Step 0**:** The below link is a documents of how to setup AWS IOT, you can refer to it for full AWS IOT knowledge.

https://docs.aws.amazon.com/iot/latest/developerguide/iot-gs.html

But, if you want to setup AzureWave AWS Connector, you would just refer to the following steps.

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

Step 1: Create AWS Account, Create an IAM user.

Please refer to the below link to setup AWS Account and IAM user.

https://docs.aws.amazon.com/iot/latest/developerguide/setting-up.html

If you have created an IAM user, please refer to the following setting to connect these two policies (AmazonFreeRTOSFullAccess, AWSIoTFullAccess) to your IAM.

https://docs.aws.amazon.com/freertos/latest/userguide/freertos-account-and-permissions.html

Step 2: Create a thing.

A thing represents a specific device or instance that can communicate with AWS IOT.

Please refer to the following link to create a thing.

https://docs.aws.amazon.com/iot/latest/developerguide/create-aws-thing.html

Step 3: Register a device

This step will create certificate and private key. You can use certificate, private key, thing name and endpoint as **AT+THING_Set** command parameter. After this command executing, the four parameter will be provision to our connector. After provisioning, you can connect to AWS IOT with MQTT or SHADOW operation.

Please refer to the steps at the following link.

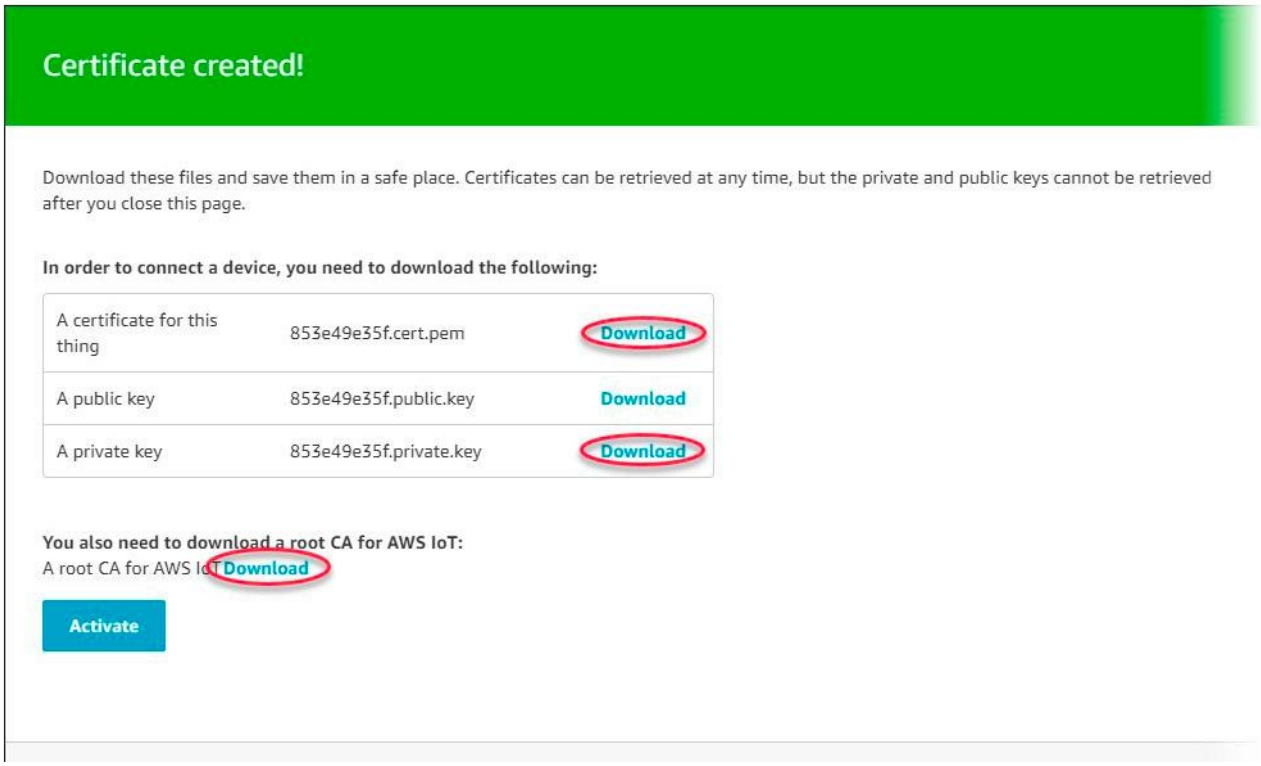
https://docs.aws.amazon.com/iot/latest/developerguide/register-device.html

After finishing the steps, please notice the following two actions:

- Download certificate and private key

In Create and activate a device certificate chapter, please download and keep the certificate and private key. Because they will be used when sending the **AT+THING_Set** command.

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

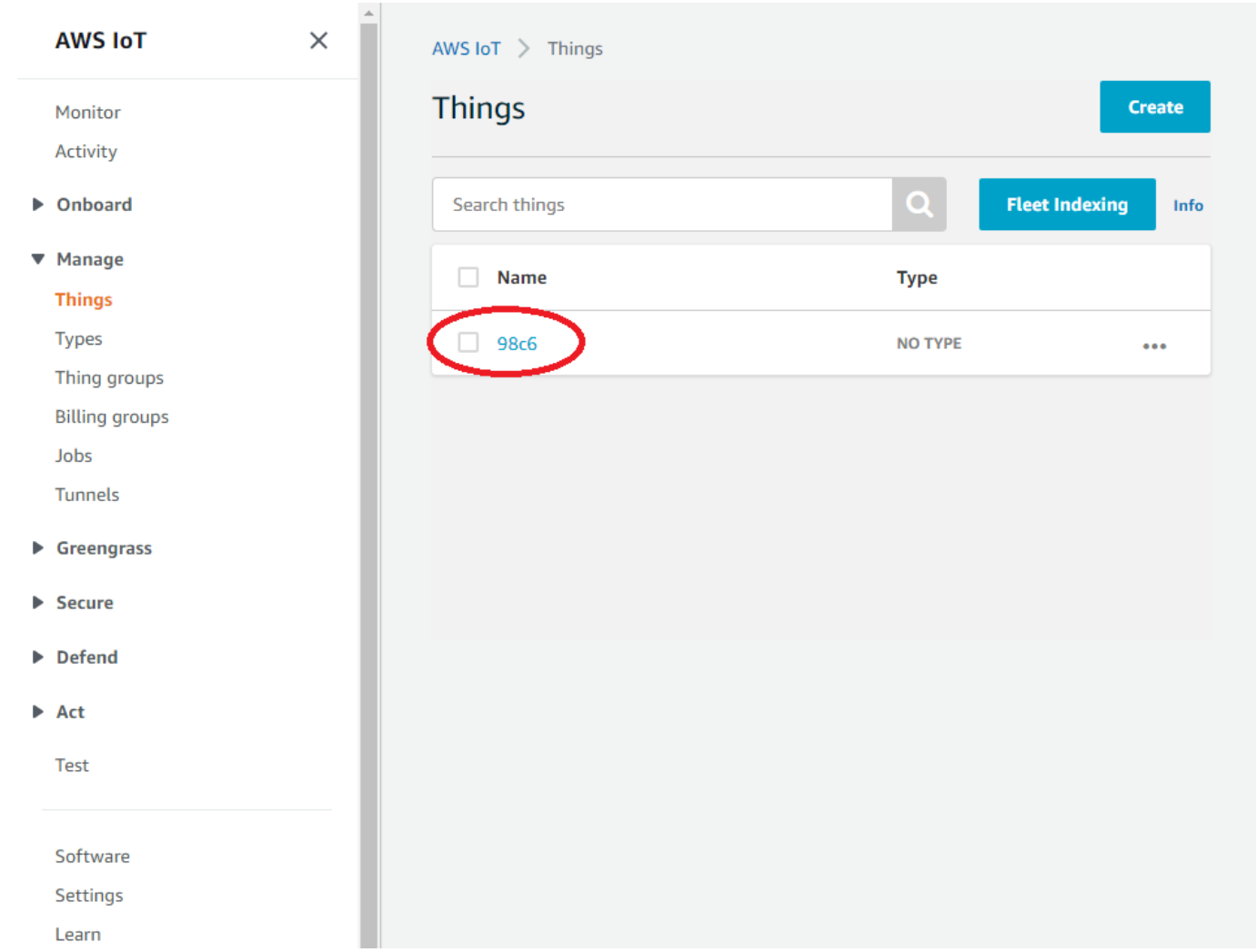

Certificate created!
Download these files and save them in a safe place. Certificates can be retrieved at any time, but the private and public keys cannot be retrieved after you close this page.
In order to connect a device, you need to download the following:
A certificate for this thing
853e49e35f.cert.pem
Download
A public key
853e49e35f.public.key
Download
A private key
853e49e35f.private.key
Download
You also need to download a root CA for AWS IoT:
A root CA for AWS IoT
Download
Activate

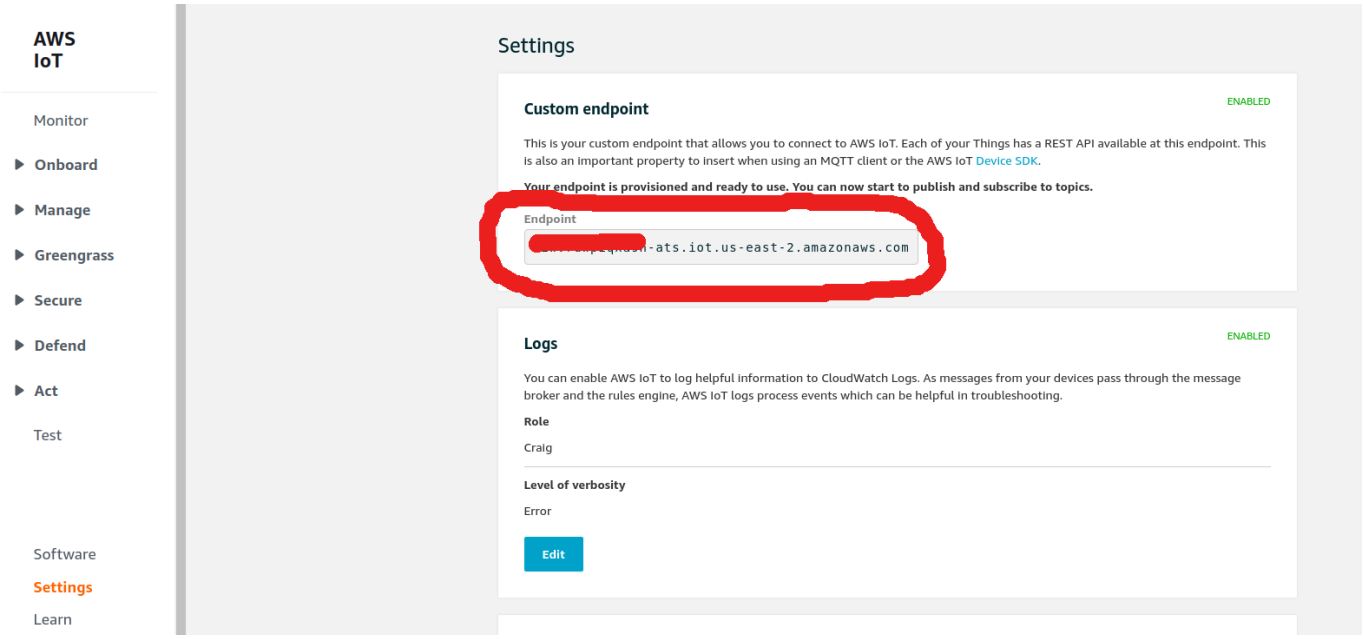
- Thing Name and Endpoint

These two data will also be used for **AT+THING_Set** command.

You can find out thing name in Manage > Things submenu, and endpoint in settings of AWS IOT Console at console.aws.amazon/iot.

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.




AWS IoT
Monitor
Onboard
Manage
Greengrass
Secure
Defend
Act
Test
Software
Settings
Learn
Settings
Custom endpoint
ENABLED
This is your custom endpoint that allows you to connect to AWS IoT. Each of your Things has a REST API available at this endpoint. This is also an important property to insert when using an MQTT client or the AWS IoT Device SDK.
Your endpoint is provisioned and ready to use. You can now start to publish and subscribe to topics.
Endpoint
-ats.iot.us-east-2.amazonaws.com
Logs
ENABLED
You can enable AWS IoT to log helpful information to CloudWatch Logs. As messages from your devices pass through the message broker and the rules engine, AWS IoT logs process events which can be helpful in troubleshooting.
Role
Craig
Level of verbosity
Error
Edit

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

2.2 Publish and Monitor MQTT message on the cloud

Step 0: Make sure the AP under test is connected to the internet using other Wi-Fi enabled devices. Assume the SSID, password and security type of the AP is MySSID, MyPassword, wpa2.

Step 1: Connect the AW-CU300A V3 to the system (refer to 2. Hardware Setup) and turn off the wireless devices near the AW-CU300A V3 (except for the device under test). Turn on the Wi-Fi module of the AW-CU300A V3 using **AT+WIFI_On**

Step 2: Check if AW-CU300A V3 is in station mode using **AT+WIFI_GetMode**

Step 3: Set and store information of the AP using **AT+WIFI_SetAP= MySSID,MyPassword,wpa2**

AT+WIFI_SetAP=**SSID**,**password,security type**
SSID: SSID of AP (case sensitive)
password: password for AP (case sensitive)
security type: OPEN | WEP | WPA | WPA2

Step 4: Connect to the AP: **AT+WIFI_Connect**

Step 5**:** Define and store Thing-specific configuration using **AT+THING_Set** command

AT+THING_Set=client ID,endpoint,client certificate,client private key
client ID: Thing name(Client ID)
endpoint: AWS IoT endpoint URL
client certificate: Certificate for this Thing (downloaded in 3.1 step 3)
client private key: Private key for this Thing (downloaded in 3.1 step 3)

You should create command as format below:

AT+THING_Set=98c6, a25giitjes2mei-ats.iot.ap-northeast-1.amazonaws.com, -----BEGIN CERTIFICATE-----\n…base64 data…\n-----END CERTIFICATE-----\n, -----BEGIN RSA PRIVATE KEY-----\n…base64 data…\n-----END RSA PRIVATE KEY-----\n

Note: a25giitjes2mei is just an example endpoint, your endpoint URL should replace it

Step 6: Connect the client to MQTT broker: **AT+MQTT_Connect**.

Step 7: Subscribe to and save MQTT topic using **AT+MQTT_Subscribe=iotdemo/1,0**

Step 8: Publish to MQTT topic using **AT+MQTT_Publish=iotdemo/1, hello, 0**

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.

AT+MQTT_Publish=topic,message,qos
topic: Topic to publish to
message: Message to publish
qos: 0 | 1

Step 9: Use the MQTT client in the AWS IoT console to monitor the messages that device sends to the AWS Cloud.

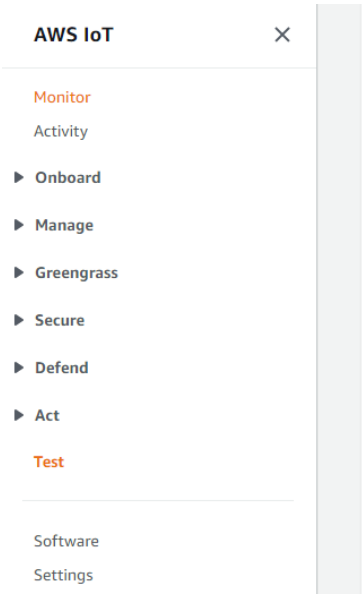
Sign in to the AWS IoT console.

https://console.aws.amazon.com/iotv2/

In the navigation pane, choose **Test** to open the MQTT client.

In **Subscription topic**, enter **iotdemo/#**, and then choose **Subscribe to topic**.

You should see the message send from device as like below.



Publish
Specify a topic and a message to publish with a QoS of 0.
iotdemo/#
Publish to t...
1 {
2 "message": "Hello from AWS IoT console"
3 }
iotdemo/1 November 06, 2020, 11:59:53 (UTC+0800) Export Hide
We cannot display the message as JSON, and are instead displaying it as UTF-8 String.
hello

The information contained herein is the exclusive property of AzureWave and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of AzureWave.